Continuous time Boolean modeling for biological signaling: application of Gillespie algorithm

Gautier Stoll (gautier.stoll@curie.fr)
Eric Viara (viara@sysra.com)
Emmanuel Barillot (emmanuel.barillot)
Laurence Calzone (laurence.calzone@curie.fr)

March 2, 2013

Abstract

This article presents an algorithm that allows modeling of biological networks in a qualitative framework with continuous time. Mathematical modeling is used as a systems biology tool to answer biological questions, and more precisely, to validate a network that describes biological observations and to predict the effect of perturbations.

There exist two major types of mathematical modeling approaches: (1) Quantitative modeling, describing various chemical species concentrations by real numbers, mainly based on differential equations and chemical kinetics formalism; (2) Qualitative modeling, describing chemical species concentrations or activities by a finite set of values, the most used being the Boolean modeling.

One major critics of qualitative modeling is that time in this approach is represented by discrete steps. Qualitative approaches, on the other hand, can efficiently describe and predict stable states, but remain inconvenient in describing transient kinetics leading to these states. In order to handle these transient events, several attempts were made to introduce continuous time in qualitative modeling. They consist in refining the time evolution in this formalism by introducing priority classes (some variables are updated with a higher priority), variable time delays, etc.

Here, we propose a modeling approach that is intrinsically continuous in time. The algorithm presented here fills the gap between qualitative and quantitative modeling. It is based on continuous time Markov process applied on a Boolean state space. In order to describe the temporal evolution, we explicitly specify the transition rates for each node. For that purpose, we built a language that can be seen as a generalization of Boolean equations. The values of transition rates have a natural interpretation: it is the inverse of the time for the transition to occur. Mathematically, this approach can be translated in a set of ordinary differential equations on probability distributions; therefore, it can be seen as an approach in between quantitative and qualitative.

We developed a C++ software, MaBoSS, that is able to simulate such a system by applying Kinetic Monte-Carlo (or Gillespie algorithm) in the Boolean state space. This software, parallelized and optimized, computes temporal evolution of probability distributions and can also estimate stationary distributions. Applications of Boolean Kinetic Monte-Carlo have been demonstrated for two qualitative models: a toy model and a published p53/Mdm2 model. Our approach allows to describe kinetic phenomena which were difficult to handle in the original models. In particular, transient effects are represented by time dependent probability distributions, interpretable in terms of cell populations.

Background

Mathematical models of signaling pathways can be seen as tools to answer biological questions. The most widely used mathematical formalisms to answer these questions are ordinary differential equations (ODEs) and Boolean modeling.

1

Ordinary differential equations (ODEs) have been widely used to model signaling pathways. It is the most natural formalism for translating detailed reaction networks into a mathematical model. Indeed, equations can be directly derived using mass action laws, Michaelis-Menten kinetics or Hill functions for each reaction in order to account for the observed behaviors. This framework has limitations, though. The first one concerns the difficulty of assigning the kinetic parameter values in the model. Ideally, these parameters would be extracted from experimental data. However, they are often chosen so as to fit qualitatively the expected phenotypes. The second limitation arises when studying cell population heterogeneity. In this case, ODEs are no longer appropriate since the approach is deterministic and thus focuses on the average behavior. To include non-determinism, an ODE model needs to be transformed into stochastic chemical model. In this formalism, a master equation is written on the probabilities of number of molecules for each species (instead of being written on ODEs of continuous concentrations). In the translation process, the same parameters used in ODEs (more particularly in ODEs written with mass action law) can be used in the master equation, but in this case, the number of initial conditions explodes along with the computation time.

Boolean formalism is another formalism used to model signaling pathways where genes/proteins are parameterized by 0s and 1s only. It is the most natural formalism to translate an influence network into a mathematical model. In such networks, each node corresponds to a species and each arrow to an interaction or an influence (positive or negative). In a Boolean model, a logical rule integrating the signs of the input links is assigned to each node. As a result, there are no real parameter values to adjust besides choosing the appropriate logical rules that best describe the system. In this paper, we will refer to a network state as a state in which each node of the influence network has a Boolean value. The set of all possible transitions between the network states is defined as a transition graph. There are two types of transition graphs, one deduced from the synchronous update strategy [1], for which all the nodes that can be updated are updated in one transition, and another one deduced from the asynchronous update strategy [2], for which only one node is updated in one transition. In the Boolean formalism, each transition can be interpreted as a time step. Real characterization of biological time is not taken into consideration. However, in many biological problems, time plays a crucial role.

As mentioned for ODE, stochasticity is an important aspect when studying cell populations, It can be done: on nodes (by randomly flipping the node state [3, 4]), on the logical rules (by allowing to change an AND gate by an OR gate [5]), and on the update rules (by defining the probability and the priority of changing one Boolean value in an asynchronous strategy [6] or by adding noise to the whole system in a synchronous strategy [7]). One of the main drawbacks of the Boolean approach is the explosion of solutions. In an asynchronous update strategy, the transition graph can reach $2^{\#nodes}$.

Both discrete and continuous frameworks have advantages and disadvantages above-mentioned. We propose here to combine some of the advantages of both approaches with what we call the "Boolean Kinetic Monte-Carlo" algorithm (BKMC). It consists of a natural generalization of the asynchronous Boolean dynamics [2], with a direct probabilistic interpretation. In BKMC framework, the dynamics is parameterized by a biological time, the order of update is noisy (less strict than priority classes introduced in GINsim [8]). A BKMC model is specified by logical rules as in regular Boolean models but with a more precise information: a numerical rate is added for each transition.

BKMC is best suited to model signaling pathways in the following cases:

- The model is based on an influence network, because BKMC is a generalization of the asynchronous Boolean dynamics. Typical examples are published models of cell cycle [6].

- The model describes processes for which information about biological time is known (relative rate, order, etc.), because in BKMC, time is parameterized by a real number. This is typically the case when studying developmental biology, where animal models provide time changes of gene/protein activities [9].

- The model describes heterogeneous cell population behavior, because BKMC has a probabilistic interpretation. For example, modeling heterogeneous cell population can help to understand tissu formation

based on cell differentiation [10].

- The model can contain many nodes (up to 64 in the present implementation), because BKMC is a simulation algorithm that converges fast. This can be useful for big models that have already been modeled with a discrete time Boolean method [11], in order to obtain a finer description of transient effects.

Previous published works have also introduced a continuous time approach in the Boolean framework([12], [13], [14], [15], [16], [17], [18]). We will review them and present BKMC approach. We will describe the C++ software, MaBoSS, developed to implement BKMC algorithm and illustrate its use with two examples, a toy model and a published model of p53-MDM2 interaction.

All abbreviations, definitions and algorithms used in this article can be found in Supplementary Material. Throughout the article, all terms that are italicized are defined in the Supplementary Material (Part 3).

Results and discussion

BKMC for continuous time Boolean model

Continuous time in Boolean modeling: past and present

In our context, we briefly recall that, in Boolean approaches for modeling networks, we define the state of each node of the network by a Boolean value (node state) and the network state by the set of node states. Any dynamics in the transition graph is represented by sequences of network states. A node state is based on the sign of the input arrows and the logic that links them. The dynamics can be deterministic (in the case of synchronized update [1]) or non-deterministic (in the case of asynchronized update [2] or probabilistic Boolean networks [7]).

The difficulty to interpret a dynamics in terms of biological time has led to several works that have generalized Boolean approaches. These approaches can be divided in two classes that we call explicit and implicit time for discrete steps.

The explicit time for discrete steps consists of adding a real parameter to each node. These parameters correspond to the time associated to each node state before it flips to another one. They need to be set for each node state ([13], [12]). Because data about these time lengths are difficult to extract from experimental studies, some works have included noise in the definition of these time parameters [18]. The drawback of this method is that the computation of the Boolean model becomes sensitive to both the type of noise and the initial conditions. As a result, these time parameters become new parameters that need to be tuned carefully and thus add complexity to the modeling.

The implicit time for discrete steps consists of adding probabilities to each transition of the transition graph, in the case of non-deterministic transitions. It is argued that these probabilities could be interpreted as adding time to a biological process. As an illustration, let us assume a small network of two nodes, A and B. At time t, A and B are inactive: [AB] = [00]. In the transition graph, there exist two possible transitions at t+1: [00] → [01] and [00] → [10]. If the first transition has a significant higher probability than the second one, then we can conclude that B has a higher tendency to activate before A. Therefore, it is equivalent to say that the activation of B is faster than the activation of A. Thus, in this case, the notion of time is implicitly modeled by setting probability transitions. In particular, *priority rules*, in the asynchronous strategy, consist of putting some of these probabilities to zero [6]. In our example, if B is faster than A then the probability of the transition [00] → [10] is zero. As a result, the prioritized nodes always activate before the others. From a different perspective but keeping the same idea, Vahedi and colleagues [14] have set up a method to deduce explicitly these probabilities from the duration of each discrete step. With the implementation of implicit time in a Boolean model, the dynamics remains difficult to interpret in terms of biological time.

As an alternative to these approaches, we propose BKMC algorithm.

Properties of BKMC algorithm

BKMC algorithm was built such as to meet the following principles:

- The state of each node is given by a Boolean number (0 or 1) (referred to as node state);

- The state of the network is given by the set of node states (referred to as network state);

- The update of a node state is based on the signs linking the incoming arrows of this node and the logic;

- Time is represented by a real number;

- Evolution is stochastic.

For that, we choose to describe the time evolution of network states by a **Markov process** with **continuous time**. Therefore, the dynamics is defined by **transition rates** inserted in a master equation (see Supplementary Material, section 1.1). Transitions for which a rate is specified are based on asynchronous Boolean dynamics.

Markov process for Boolean model

In BKMC, we adapt the Markov process to the Boolean approach.

Consider a network of n nodes (or agents, that can represent any species, *i.e.* mRNA, proteins, complexes, etc.). In a Boolean framework, the network state of the system is described by a vector $\mathbf{S}$ of Boolean values, i.e. $S_i \in \{0, 1\}, i = 1, \ldots, n$ where S_i is the *state of the node* i. The set of all possible network states, also referred to as the *network state space*, will be called Σ.

A stochastic description of state evolution is represented by a *stochastic process* $s(t)$ defined on $t \in I \subset \mathbb{R}$ to the network state space, where I is an interval: for each time $t \in I \subset \mathbb{R}$, $s(t)$ represents a random variable to the network state space:

$$
\begin{aligned}
\mathbf{P}\left[s(t) = \mathbf{S}\right] &\in& [0, 1] \text{ for any state } \mathbf{S} \in \Sigma \\
\sum_{\mathbf{S} \in \Sigma} \mathbf{P}\left[s(t) = \mathbf{S}\right] &=& 1
\end{aligned}
\tag{1}
$$

Notice that the random variables $s(t)$ are not independent, therefore $\mathbf{P}\left[s(t) = \mathbf{S}, s(t') = \mathbf{S}'\right] \neq \mathbf{P}\left[s(t) = \mathbf{S}\right] \mathbf{P}\left[s(t') = \mathbf{S}'\right]$. From now on, we define $\mathbf{P}\left[s(t) = \mathbf{S}\right]$ as *instantaneous probabilities* (or first order probabilities). Since the instantaneous probabilities do not define the full stochastic process, all possible joint probabilities should also be defined.

In order to simplify the stochastic process, Markov property is imposed. It can be expressed in the following way: "the conditional probabilities in the future, related to the present and the past, depend only on the present" (see Supplementary Material, section 1.1 for the mathematical definition). A stochastic process with the Markov property is called a *Markov process*.

Any Markov process can be defined by (see Van Kampen [19], chapter IV):

1. an initial condition:

$$
\mathbf{P}\left[s(0) = \mathbf{S}\right] \; ; \forall \mathbf{S} \in \Sigma
\tag{2}
$$

2. conditional probabilities (of a single condition):

$$
\mathbf{P}\left[s(t) = \mathbf{S} | s(t') = \mathbf{S}'\right] \; ; \forall \mathbf{S}, \mathbf{S}' \in \Sigma \; ; \forall t', t \in I; t' < t
\tag{3}
$$

Concerning time, two cases can be considered:

- Time is discrete: $t \in I = \{t_0, t_1, \cdots\}$. In that case, it can be shown [20] that all possible conditional probabilities are function of *transition probabilities*: $\mathbf{P}\left[s(t_i) = \mathbf{S}|s(t_{i-1}) = \mathbf{S}'\right]$. In that case, a Markov process is often named a Markov chain.

- Time is continuous: $t \in I = [a, b]$. In that case, it can be shown [19] that all possible conditional probabilities are function of *transition rates*: $\rho_{(\mathbf{S}' \to \mathbf{S})}(t) \in [0, \infty[$ (see Supplementary Material, section 1.1 for the definition of transition rates).

Notice that a discrete time Markov process can be derived from continuous time Markov process, called *Jump Process*, with the following transition probabilities:

$$\mathbf{P}_{\mathbf{S} \to \mathbf{S}'} \equiv \frac{\rho_{\mathbf{S} \to \mathbf{S}'}}{\sum_{\mathbf{S}''} \rho_{\mathbf{S} \to \mathbf{S}''}}$$

If the transition probabilities or transition rates are time independent, the Markov process is called a *time independent Markov process*. In BKMC, only this case will be considered. For a time independent Markov process, the *transition graph* (often called Boolean state graph in the Boolean framework) can be defined as follows: a transition graph is a graph in Σ, with an edge between $\mathbf{S}$ and $\mathbf{S}'$ if and only if $\rho_{\mathbf{S} \to \mathbf{S}'} > 0$ (or $\mathbf{P}\left[s(t_i) = \mathbf{S}|s(t_{i-1}) = \mathbf{S}'\right] > 0$ if time is discrete).

Asynchronous Boolean dynamics as a discrete time Markov process

Asynchronous Boolean dynamics [2] is widely used in Boolean modeling. It can be easily interpreted as discrete time Markov process [21, 22] as shown below.

In the case of asynchronous Boolean dynamics, the system is given by n nodes (or agents), with a set of directed arrows linking these nodes and defining a network. For each node i, a Boolean *logic* $B_i(\mathbf{S})$ is specified that depends only on the nodes j for which there exists an arrow from node j to i (e.g. $B_1 = S_3$ AND NOT S_4, where S_3 and S_4 are the Boolean values of nodes 3 and 4 respectively). The notion of *asynchronous transition* (AT) can be defined as a pair of network states $(\mathbf{S}, \mathbf{S}') \in \Sigma$, written $(\mathbf{S} \to \mathbf{S}')$ such that

$$\begin{aligned} S'_j &= B_j(\mathbf{S}) \text{ for a given } j \\ S'_i &= S_i \text{ for } i \neq j \end{aligned} \tag{4}$$

To define a Markov process, the transition probabilities $\mathbf{P}\left[s(t_i) = \mathbf{S}|s(t_{i-1}) = \mathbf{S}'\right]$ can be defined such that: given two network states $\mathbf{S}$ and $\mathbf{S}'$, let $\gamma(\mathbf{S})$ be the number of asynchronous transitions from $\mathbf{S}$ to all possible states $\mathbf{S}'$. Then

$$\begin{aligned} \mathbf{P}\left[s(t_i) = \mathbf{S}'|s(t_{i-1}) = \mathbf{S}\right] &= 1/\gamma(\mathbf{S}) \text{ if } (\mathbf{S} \to \mathbf{S}') \text{ is an AT} \\ \mathbf{P}\left[s(t_i) = \mathbf{S}'|s(t_{i-1}) = \mathbf{S}\right] &= 0 \text{ if } (\mathbf{S} \to \mathbf{S}') \text{ is not an AT} \end{aligned} \tag{5}$$

In this formalism, asynchronous Boolean dynamics completely defines a discrete time Markov process when the initial condition is specified. Notice that here the transition probabilities are time independent, i.e. $\mathbf{P}\left[s(t_i) = \mathbf{S}|s(t_{i-1}) = \mathbf{S}'\right] = \mathbf{P}\left[s(t_{i+1}) = \mathbf{S}|s(t_i) = \mathbf{S}'\right]$. Other definition of $\gamma(\mathbf{S})$ can be used, defining another Markov process that have the same transition graph. Therefore, the approaches mention above, that introduce time implicitly by adding probabilities to each transition of the transition graph, can be seen as a generalization of the definition of $\gamma(\mathbf{S})$.

Continuous time Markov process as a generalization of asynchronous Boolean Dynamics

To transform the discrete time Markov process described above in a continuous time Markov process, transition probabilities should be replaced by transition rates $\rho_{(\mathbf{S} \to \mathbf{S}')}$. In that case, conditional probabilities are computed by solving a *master equation* (equation 2 in Supplementary Material, section 1.1). We present below the corresponding numerical algorithm (*Kinetic Monte-Carlo* [23]).

Because we want a generalization of asynchronous Boolean dynamics, transition rates $\rho_{(\mathbf{S} \to \mathbf{S}')}$ are non-zero <u>only if</u> $\mathbf{S}$ and $\mathbf{S}'$ differ by only one node. In that case, each Boolean logic $B_i(\mathbf{S})$ is replaced by two functions $R_i^{\text{up/down}}(\mathbf{S}) \in [0, \infty[$. The transition rates are defined as follow: if i is the node that differs from $\mathbf{S}$ and $\mathbf{S}'$,

$$
\begin{aligned}
\rho_{(\mathbf{S} \to \mathbf{S}')} &= R_i^{\text{up}}(\mathbf{S}) \text{ if } S_i = 0 \\
\rho_{(\mathbf{S} \to \mathbf{S}')} &= R_i^{\text{down}}(\mathbf{S}) \text{ if } S_i = 1
\end{aligned}
\tag{6}
$$

R_i^{up} corresponds to the activation rate of node i, R_i^{down} corresponds to the inhibition rate of node i. Therefore, the continuous Markov process is completely defined by all these $R^{\text{up/down}}$ and an initial condition.

Asymptotic behavior of continuous time Markov process

In the case of continuous time Markov process, instantaneous probabilities always converge to a stationary distribution (see Supplementary Material, corollary 2, section 1.2). A *stationary distribution* of a given Markov process corresponds to the set of instantaneous probabilities of a stationary Markov process which has the same transition probabilities (or transition rates) of the given discrete (or continuous) time Markov process. A *stationary Markov process* has the following property: for every joint probability $\mathbf{P}\left[s(t_1) = \mathbf{S}^{(1)}, s(t_2) = \mathbf{S}^{(2)}, \ldots \right]$ and $\forall \tau$,

$$
\mathbf{P}\left[s(t_1) = \mathbf{S}^{(1)}, s(t_2) = \mathbf{S}^{(1)}, \ldots \right] = \mathbf{P}\left[s(t_1 + \tau) = \mathbf{S}^{(1)}, s(t_2 + \tau) = \mathbf{S}^{(1)}, \ldots \right]
\tag{7}
$$

Notice that instantaneous probabilities $\mathbf{P}\left[s(t) = \mathbf{S} \right]$ of a stationary stochastic process are time independent.

The asymptotic behavior of a continuous time Markov process can be detailed by using the concept of indecomposable stationary distributions: *indecomposable stationary distributions* are stationary distributions that cannot be expressed as linear combination of different stationary distributions. Notice that a linear combination of stationary distributions is also a stationary distribution, up to a constant. This comes from the fact that instantaneous probabilities are solutions of a master equation, which is linear (see Supplementary Material, equation 2, section 1.1). Therefore, a complete description of the asymptotic behavior is given by the linear combination of indecomposable stationary distributions, to which the Markov process converges.

Oscillations and cycles

In order to describe a periodic behavior, the notion of cycle and oscillation for a continuous time Markov process is defined precisely.

A *cycle* is a loop in the transition graph. This is a topological characterization that does not depend on the exact value of transition rates. It can be shown that a cycle with no outcoming edge corresponds to an indecomposable stationary distribution (see Supplementary Material, corollary 1, section 1.2).

The question is then to link the notion of cycle to that of periodic behavior of instantaneous probabilities. Instantaneous probabilities cannot be perfectly periodic; at most, they have a damped ocscillating behavior (see Supplementary Material, section 1.3). Let us define formally a *damped oscillatory* Markov process as a continuous time process that has at least one instantaneous probability with an infinite number of extrema.

According to theorems described in Supplementary Material (theorems 6-8 and Corollary 3, section 1.3), a necessary condition for having damped oscillation is that the transition matrix (see Supplementary Material, equation 4, section 1.1) has at least one non-real eigenvalue. In that case, there always exists an initial condition that produces damped oscillations. For the transition matrix to have a non-real eigenvalue, a Markov process needs to have a cycle. However, the reverse is not true. In the toy model of single cycle, presented in the section of examples, non-real eigenvalues may or may not exist, according to different sets of transition rates, although the transition graph remains the same.

BKMC: Kinetic Monte-Carlo (Gillespie algorithm) applied to continuous time asynchronous Boolean Dynamics

It has been previously stated that a continuous time Markov process is completely defined by its initial condition and its transition rates. For computing any conditional probability (and any joint probability), a set of linear differential equations has to be solved (the *master equation*). Theoretically, the master equation can be solved exactly by computing the exponential of the transition matrix (see Supplementary Material, equation 5, section 1.1). However, because the size of this transition matrix is $2^n \times 2^n$, practical computation soon becomes impossible if n is large. To remedy this problem, it is possible to use a simulation algorithm that samples the probability space by computing time trajectories in the network state space.

The Kinetic Monte-Carlo [23] (or Gillespie algorithm [24]) is a simple algorithm for exploring the probability space of a Markov process defined by a set of transition rates. In fact, it can be understood as a formal definition of a continuous time Markov process. This algorithm produces a set of *realizations* or *stochastic trajectories* of the Markov process, given a set of uniform random numbers in $[0, 1[$. By definition, a trajectory $\hat{\mathbf{S}}(t)$ is a function from a time window $[0, t_{\max}]$ to Σ. The set of *realizations* or *stochastic trajectories* represents the given Markov process in the sense that these trajectories can be used to computed probabilities. Practically, a finite set of these trajectories is produced, then probabilities are estimated from this finite set (as described below). The algorithm is based on an iterative step: from a state $\mathbf{S}$ at time t_0 (given two uniform random numbers), it produces a transition time δt and a new state $\mathbf{S}'$, with the following interpretation: the trajectory $\hat{\mathbf{S}}(t)$ is such that $\hat{\mathbf{S}}(t) = \mathbf{S}$ for $t \in [t_0, t_0 + \delta t[$ and $\hat{\mathbf{S}}(t_0 + \delta t) = \mathbf{S}'$. Iteration of this step is done until a specified maximum time is reached. The initial state of each trajectory is based on the (probabilistic) initial condition, that needs also to be specified.

The exact iterative step is the following, given $\mathbf{S}$ and two uniform random number $u, u' \in [0, 1[$:

1. Compute the total rate of possible transitions for leaving $\mathbf{S}$ state:
   $\rho_{\text{tot}} \equiv \sum_{\mathbf{S}'} \rho_{(\mathbf{S} \to \mathbf{S}')}.$

2. Compute the time of the transition: $\delta t \equiv -\log(u)/\rho_{\text{tot}}$

3. Order the possible new states $\mathbf{S}'^{(j)}, j = 1 \ldots$ and their respective transition rates $\rho^{(j)} = \rho_{(\mathbf{S} \to \mathbf{S}'^{(j)})}$.

4. Compute the new state $\mathbf{S}'^{(k)}$ such that $\sum_{j=0}^{k-1} \rho_j < (u' \rho_{\text{tot}}) \leq \sum_{j=0}^{k} \rho_j$ (by convention, $\rho^{(0)} = 0$).

The application of this algorithm to continuous time Markov process in network state space will be referred to as *Boolean Kinetic Monte-Carlo* or BKMC.

Practical use of BKMC, through MaBoSS tool

Biological data are summarized into an influence network with logical rules associated to each node of the network. The value of one node depends on the value of the input nodes. For BKMC, another layer of information is provided when compared to the standard definition of Boolean models: transition rates are provided for all nodes, specifying the rates at which the node turns on and off based on their logic for both the on and off rules. This refinement conserves the simplicity of Boolean description but allows to reproduce the observed biological dynamics. The parameters do not need to be exact as in nonlinear ordinary differential equation models but they can be used to illustrate the relative speed of reactions. For that purpose, we developed a software tool, MaBoSS, that applies BKMC algorithm. MaBoSS stands for Markov Boolean Stochastic Simulator. Practically, MaBoSS needs two input files: one describing the network and its transition rates, one describing the parameters controlling the different estimates described in the Methods section. Source code, reference card and examples are available on the web: https://maboss.curie.fr.

Transition rates with MaBoSS language: modeling biological processes

MaBoSS defines transition rates $\rho_{(\mathbf{S} \to \mathbf{S}')}$ by the functions $R_j^{\text{up/down}}(\mathbf{S})$ (see equation 6). The format of these functions is very flexible. It includes all Boolean operators (AND, OR, NOT, XOR), arithmetic operators (+,-

,* /), external variables, node variables, comparison operators and the conditional operator (?:). Examples of the use of the language are given below to illustrate three different cases: different speeds for different inputs, buffering effect and the translation of discrete variables (with more than the 2 values, 0 and 1) in MaBoSS.

- Modeling different speeds of incoming influences: suppose that C is activated by A and B, but that B can activate C faster than A. In this case, we write:

```
node C {
rate_up= B ? $kb : (A ? $ka : 0.0);
rate_down= (A | B ) ? 0.0 : 1.0}
```

When C is off (equal to 0), C is activated by B at a speed kb. If B is absent, then C is activated by A at a speed $ka. If both are absent, C is not activated. Note that if both A and B are present, because of the way the logic is written in this particular case, C is activated at a speed $kb. When C is on (equal to 1), C is inactivated at a rate equal to 1 if A and B are both absent.

To implement the synergetic effect of A and B, *i.e.* when both A and B are on, C activates at a rate $kab, then we can write:

```
node C {
rate_up= (A & !B ? $ka : 0)+(B & !A ? $kb : 0) + (A & B ? $kab : 0.0);
rate_down= (A | B ) ? 0.0 : 1.0}
```

- Modeling buffering effect: suppose that B is activated by A, but that B can remain active a long time after A has shut down. For that, it is enough to define different speeds of activation and inhibition:

```
node B {
rate_up= A ? 2.0 : 0.0;
rate_down= A ? 0.0 : 0.001;}
```

B is activated by A at a rate equal to 2. When A is turned off, B is inactivated more slowly at a rate equal to 0.001.

- Modeling more than two discrete states for a given node: Suppose that B is activated by A, but if the activity of A is maintained, B can reach a second level. For this, we define a second node B_h (for "B high") with the following rules:

```
node B {
rate_up= A ? 1.0 : 0.0;
rate_down= (A | B_h) ? 0.0 : 1.0;}
```

```
node B_h {
rate_up= (A & B) ? 1.0 : 0.0;
rate_down= (A) ? 0.0 : 1.0;}
```

In this example, B is separated in two variables: B which corresponds to the first level of B and B_h which corresponds to the higher level of B. B is activated by A at a rate equal to 1. If A disappears before B has reached its second level B_h then B is turned off at a rate equal to 1. If A is maintained and B is active, then B_h is activated at a rate equal to 1. When A is turned off, B_h is inactivated at a rate equal to 1.

Simulation input parameters in MaBoSS

To simulate a process in MaBoSS, a set of parameters need to be adjusted (see simulation parameters in reference card). MaBoSS assigns default values, however, they need to be tuned for each model to achieve optimal performances: best balance between the convergence of estimates and the computation time. Therefore, several simulations should be run with different set of parameters for best tuning.

- Internal nodes: *node.is_internal* As explained in Methods ("Initial conditions and outputs"), internal nodes correspond to species that are not measured explicitly. Practically, the higher the number of internal nodes, the better the convergence of the BKMC algorithm.

- Time window for probabilities: *timetick*

  This parameter is used to compute estimates of network state probabilities (see "Network state probabilities on time window" in Methods). A time window can be set as the minimum time needed for nodes to change their states. This parameter also controls the convergence of probability estimates. The larger the time window parameter, the better the convergence of probability estimates. With practice, the tradeoff between timetick parameter value and the convergence speed will be defined.

- Maximum time: *max_time*

  MaBoSS produces trajectories for a predefined amount of time set by the parameter: max_time. This maximum time needs to be specified. If the time of the biological process is known, then the maximum time parameter can be explicitly set. If the time of the biological process is not known, then there exists a more empirical way to set the maximum time. It is advised to choose a maximum time parameter that is slightly bigger than the inverse of the smallest transition rate.

  Note that the computing time in MaBoSS is proportional to this maximum time. Moreover, the choice of the maximum time impacts the stationary distribution estimates (see below): a longer maximum time increases the quality of these estimates.

- Number of trajectories: *sample_count*

  This parameter directly controls the quality of BKMC estimation algorithm. Practically, the convergence of the estimates increases as the number of trajectories is increased.

- Number of trajectories (*statdist_traj_count*) and similarity threshold (*statsdist_cluster_threshold*) for stationary distribution estimates

  The *statdist_traj_count* parameter corresponds to a subset of trajectories use only for stationary distribution estimates (the *statdist_traj_count* first trajectories are chosen by the algorithm). To avoid explosion of computing time, this parameter needs to be lower than the number of trajectories (rather than equal to).

  The *statsdist_cluster_threshold* parameter corresponds to the threshold for constructing the clusters of stationary distribution estimates. Ideally, it should be set to a high value (close to 1). However, if the threshold is too high then the clustering algorithm might not be efficient.

  For optimal results, the identification of the full set of indecomposable stationary distributions should be done in the following way:

  - Run a simulation with an initial condition and maximum time, with a reasonable similarity threshold (around 0.8) and a number of trajectories (around 1000). As a response, MaBoSS provides a set of indecomposable stationary distributions (it corresponds to the stationary distributions associated to each cluster).
  - Select the states with non-zero probability in the set of indecomposable stationary distributions and set them as initial conditions, increase the maximum time (*max_time*), the number of trajectories (*statdist_traj_count*) and/or the similarity threshold (*statsdist_cluster_threshold*).

– Run the simulations for these new parameters.

– If new indecomposable stationary distributions appear, start again the two previous steps. Stop when the indecomposable stationary distributions remain stable with respect to the simulation parameters, *i.e.* after several rounds.

Comparison with biological data

The relationship between a model and experimental data is strongly dependent on the type of model and the question that needs to be solved.

Because MaBoSS is based on Boolean modeling, the biological data need to be discretized. Each node of the model should represent discrete levels of the respective species (mRNA, protein, protein complex, etc.). It is possible to have more than two discrete levels in a model, as shown in the example "Modeling more than two discrete states for a given node".

The transitions rates are positive numbers that should be introduced in a model; it is possible to extract them from experimental data, using the following property: the rate of a given transition is the inverse of the mean time, for this transition to happen. It should be noticed than BKMC is an algorithm based on a linear equation (Supplementary Material, equation 2, section 1.1); therefore, small variations of transition rates won't affect qualitative behavior of a model.

The basic outputs of BKMC algorithm are network state probabilities over time. These can be interpreted in terms of a cell population, once experimental data are discretized. The asymptotic behavior of a model, represented by a linear combination of indecomposable stationary distributions, can be interpreted as a combination of cell sub-populations. More precisely, consider an indecomposable stationary distribution and the associated set of network states with non-zero probability. A cell in such a network state can only evolve in other network states with non-zero probability, within the same indecomposable stationary distribution (Supplementary Material, corollary 1, section 1.2 and by using the definition of strongly connected component with no outcoming edge). Therefore, the set of network states with non-zero probability can be interpreted as a sub-population whose cells evolve only within the sub-population.

Examples

Two models using BKMC applied to Boolean networks are given as examples. The first one is a toy model, illustrating the dynamics of a single cycle. The second one uses a published Boolean model of p53-Mdm2 response to DNA damage. Note that MaBoSS has been used for these two examples, but Markov process can be computed directly, without our BKMC algorithm because the model is small enough (by computing exponential of transition matrix, see Supplementary Material, section 1.1), as proposed in [16]. BKMC is best suited for larger networks, when the network state space is too large to be computed with standard existing tools ($>\sim 2^{10}$). These examples were chosen for their simplicity, and because they illustrate how global characterizations (entropy and transition entropy, see "Entropies" in Methods) can be used.

For the purpose of this article, we built the transition graphs for both examples (with GINsim [8]) in order to help the reasoning. However, BKMC algorithm does not construct the transition graph explicitly.

Toy model of a single cycle

We consider three species, A, B and C, where A is activated by C and inhibited by B, B is activated by A and C is activated by A or B (Figure 1).

The model is defined within the language of MaBoSS (defined in the web page, https://maboss.curie.fr). The associated transition graph is shown in figure 2. In this model, the only stationary distribution is the fixed point [ABC]=[000]. We studied two cases: when the rate of the transition from state [001] to state [000] is either fast or slow (inactivation of C). We will refer to this transition rate as the *escape rate*.

In the first case, when the escape rate is fast, we set the parameter for the transition to a high value (rate_up = 10). In figure 3, we notice that the probability that [ABC] is equal to [000] converges to 1. We
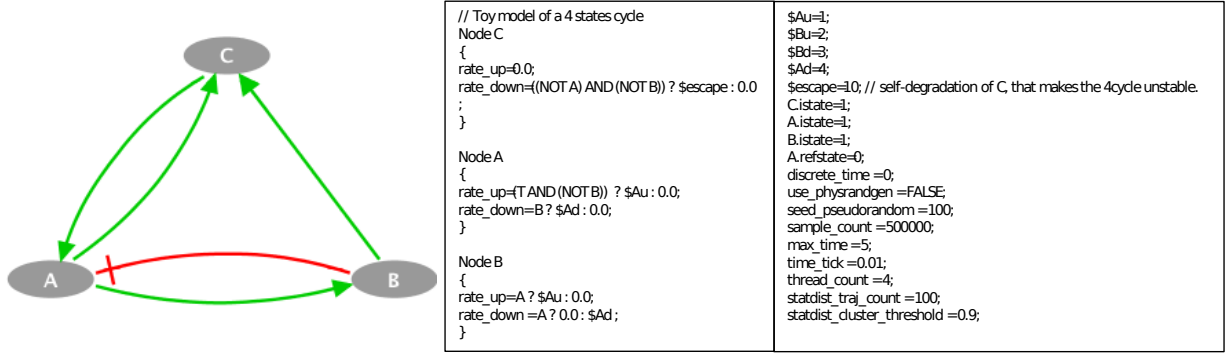
```
// Toy model of a 4 states cycle          $Au=1;
Node C                                     $Bu=2;
{                                          $Bd=3;
rate_up=0.0;                               $Ad=4;
rate_down=((NOT A) AND (NOT B)) ? $escape : 0.0   $escape=10; // self-degradation of C, that makes the 4cycle unstable.
;                                          C.istate=1;
}                                          A.istate=1;
                                           B.istate=1;
Node A                                     A.refstate=0;
{                                          discrete_time =0;
rate_up=(T AND (NOT B)) ? $Au : 0.0;       use_physrandgen =FALSE;
rate_down=B ? $Ad : 0.0;                   seed_pseudorandom =100;
}                                          sample_count =500000;
                                           max_time =5;
Node B                                     time_tick =0.01;
{                                          thread_count =4;
rate_up=A ? $Au : 0.0;                     statdist_traj_count =100;
rate_down =A ? 0.0 : $Ad ;                 statdist_cluster_threshold =0.9;
}
```

Figure 1: Toy model of a single cycle. (A) Influence network. (B) Logic and transition rates of the model. (C) Simulation parameters
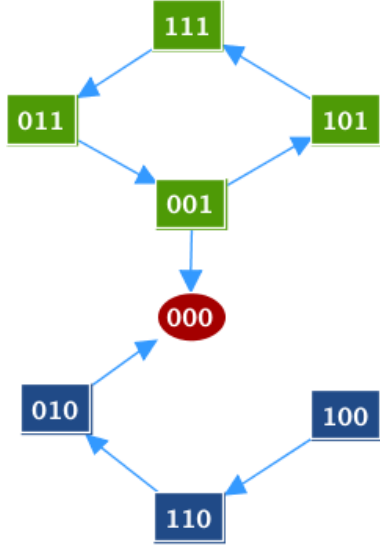
Figure 2: Transition graph for the toy model. The node state should be read as [ABC] = [***]. [ABC]=[100] corresponds to a state in which only A is active. The nodes in green belong to a cycle, the node in red is the fixed point and the other state nodes are in blue.

can conclude that [ABC]=[000] is a fixed point. In addition, the entropy and the transition entropy converge to zero. With BKMC, these properties correspond to a signature of a fixed point. The peak in the entropy (between times 0 and 0.6) corresponds to a set of transiently activated states before reaching the fixed point.

In the second case, when the escape rate is slow, we set the parameter for the transition to a low value (rate_down = 10^{-5}). The model seems to show another stationary distribution. As illustrated in figure 4, the transition entropy is and remains close to zero but the entropy does not converge to zero, which is the signature of a cyclic stationary distribution (see "Entropies" in Methods). This corresponds to the cycle [111] $\rightarrow$ [011] $\rightarrow$ [001] $\rightarrow$ [101] in the transition graph (2). However, as seen in the transition graph, one state in the cycle will eventually lead the fixed point (through the transition [001] $\rightarrow$ [000] in figure 2). Therefore, if the temporal evolution is plotted on a larger time scale (Figure 5), it looks similar to the case of fast escape rate. This case can be anticipated. Indeed, the value of the transition entropy of figure 4 is
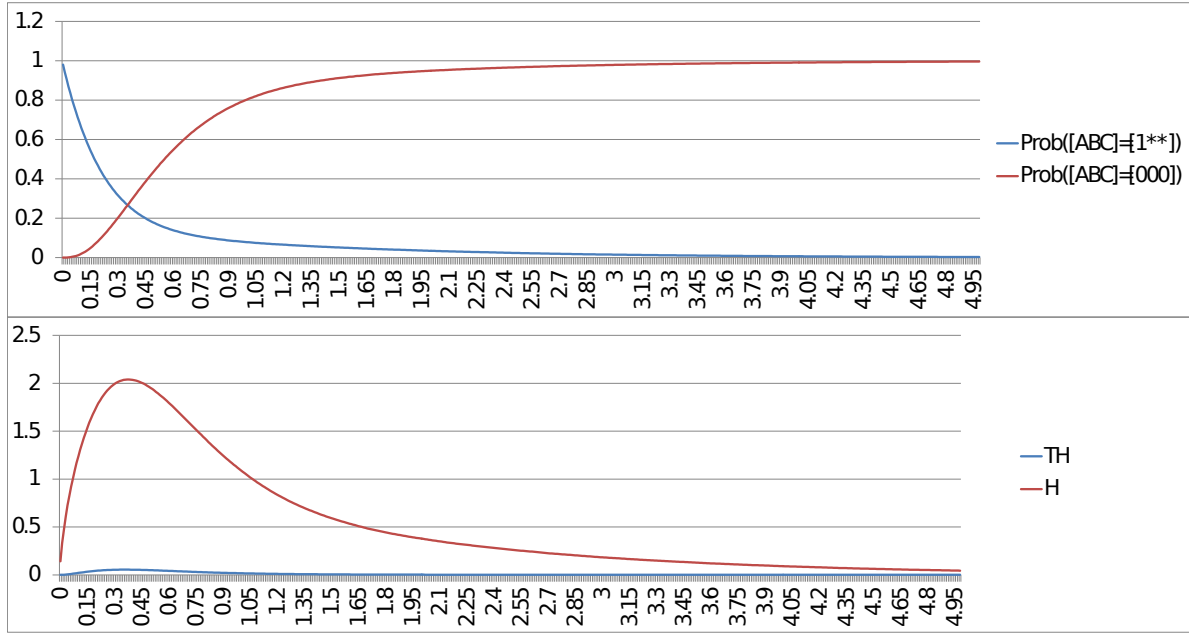
Figure 3: BKMC algorithm applied to the toy model, with a fast escape rate. Time trajectory of probabilities ([ABC]=[000] and [ABC]=[1**] where * can be either 0 or 1), the entropy (H) and the transition entropy (TH) are plotted. Because the probability of [ABC]=[000] converges to 1, [ABC]=[000] is a fixed point. The asymptotic behavior of both the entropy and the transition entropy is also the signature of a fixed point.

not exactly zero, but 10^{-4}. Therefore, the cyclic behavior is not stable. We can conclude on stable cyclic behaviors only when the transition entropy is exactly zero.

By considering the spectrum of the transition matrix (see Supplementary Material, section 1.1 and proof of theorem 4), it can be proven that the model with a slow escape rate is a damped oscillatory process whereas the model with a large escape rate is not a damped oscillatory process. As mentioned previously, a cycle in the transition graph may or may not lead to an oscillatory behavior. Moreover, if the transition entropy seems to converge to a small value on a small time scale, and the entropy does not, this behavior illustrates the case of a transient cycle in the transition graph.

p53-Mdm2 signaling

We consider a model of p53 response to DNA damage [18]. p53 interacts with Mdm2, which appears in two forms, cytoplasmic and nuclear. On one hand, p53 upregulates the level of cytoplasmic Mdm2 which is then transported into the nucleus and inhibits the export of nuclear Mdm2. On the other hand, Mdm2 facilitates the degradation of p53 through ubiquitination. In the model, stress regulates the level of DNA damage, which in turn participates in the degradation process of Mdm2. p53 inhibits the DNA damage signal by promoting DNA repair. Here, stress is not shown explicitly (Figure 6).

The model is defined within the language of MaBoSS, with two levels of p53 as it is done in Abou-Jaoudé *et al.* [18]. The model is implemented in MaBoSS (provided in the web page (https://maboss.curie.fr) along with the simulation parameters). The associated transition graph is given in figure 7. It shows the existence of two cycles and of a fixed point [p53 Mdm2C Mdm2N Dam] = [0010].

In order to represent the activity of p53, time evolution of its expectation value is shown in figure 8, with the initial condition: [p53 Mdm2C Mdm2N Dam] = [0*11]. The activation of p53 seems to be transient.

The qualitative results obtained with MaBoSS are similar to those of Abou-Jaoudé and colleagues. However, at the level of cell population, some discrepancies appear: in figure 8, no damped oscillations can
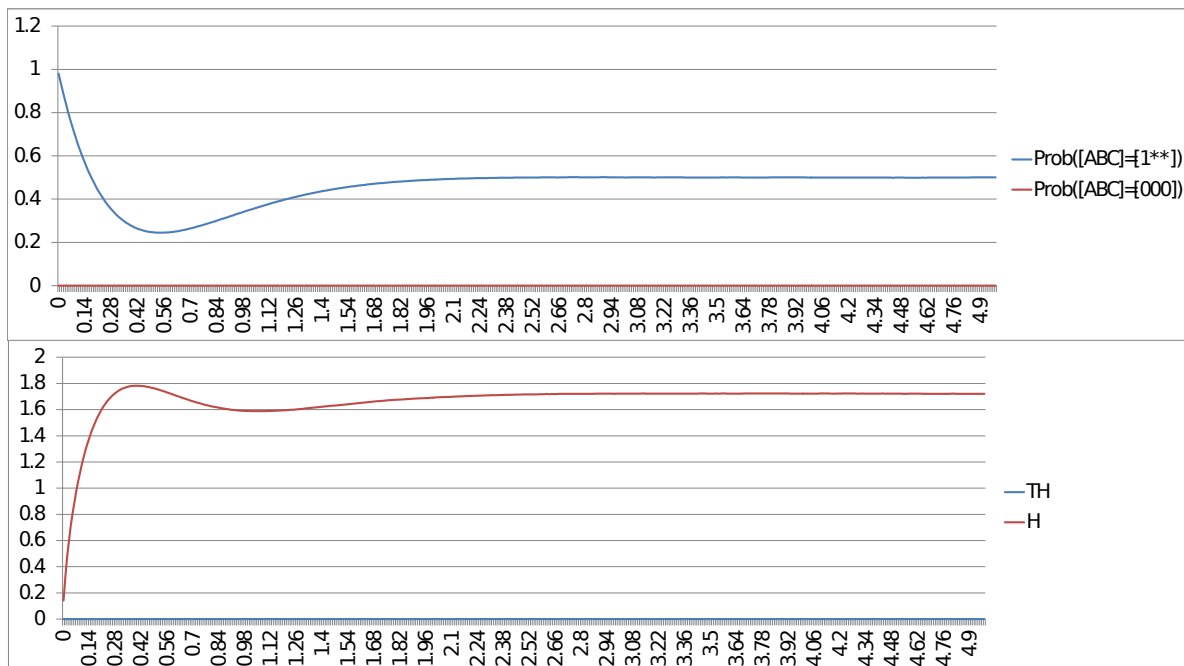
Figure 4: BKMC algorithm applied to the toy model, with a slow escape rate. Time trajectory of probabilities ([ABC]=[000] and [ABC]=[1**]), the entropy (H) and the transition entropy (TH) are plotted. The asymptotic behavior of both the entropy and the transition entropy seems to be the signature of a cycle.

be seen as opposed to figure 8 of their article. The reason is that in their computations, noise imposed on time is defined by a square distribution on a limited time frame, whereas in BKMC, Markovian hypotheses imply that the noise distribution is more spread out from 0 to infinity. The consequence is that synchronization is lost very fast. Damped oscillations could be observed with BKMC with a particular set of parameters: fast activation of p53 and slow degradation of p53 (results not shown).

With MaBoSS, we clearly interpret the system as a population and not as a single cell. In addition, we can simulate different contexts, presented in the initial article as different models, within one model that used different parameters to account for these contexts (see web page https://maboss.curie.fr for more details).

Note that the existence of transient cycles, as shown in the toy model, can be deduced from the time trajectory of the entropy that is significantly higher than the time trajectory of the transition entropy (which is non zero therefore the transient cycles are not stable) (Figure 9). An other indirect effect of transient cycles is the flat part in p53 standard deviation, in figure 8.

Conclusions

In this work, we present a new algorithm, Boolean Kinetic Monte-Carlo or BKMC, applicable to dynamical simulation of signaling networks based on continuous time in the Boolean framework. BKMC algorithm is a natural generalization of asynchronous Boolean dynamics [2], with time trajectories that can be interpreted in terms of biological time. The variables of the Boolean model are biological species and the parameters are rates of activation or inhibition of these species. These parameters can be deduced from experimental data.We applied this algorithm to two different models: a toy model that illustrates a simple cyclic behavior, and a published model of p53 response to DNA damage.

This algorithm is provided within a freely available software, MaBoSS, that can run BKMC algorithm on networks up to 64 nodes, in the present version. The construction of a model uses a specific grammar that
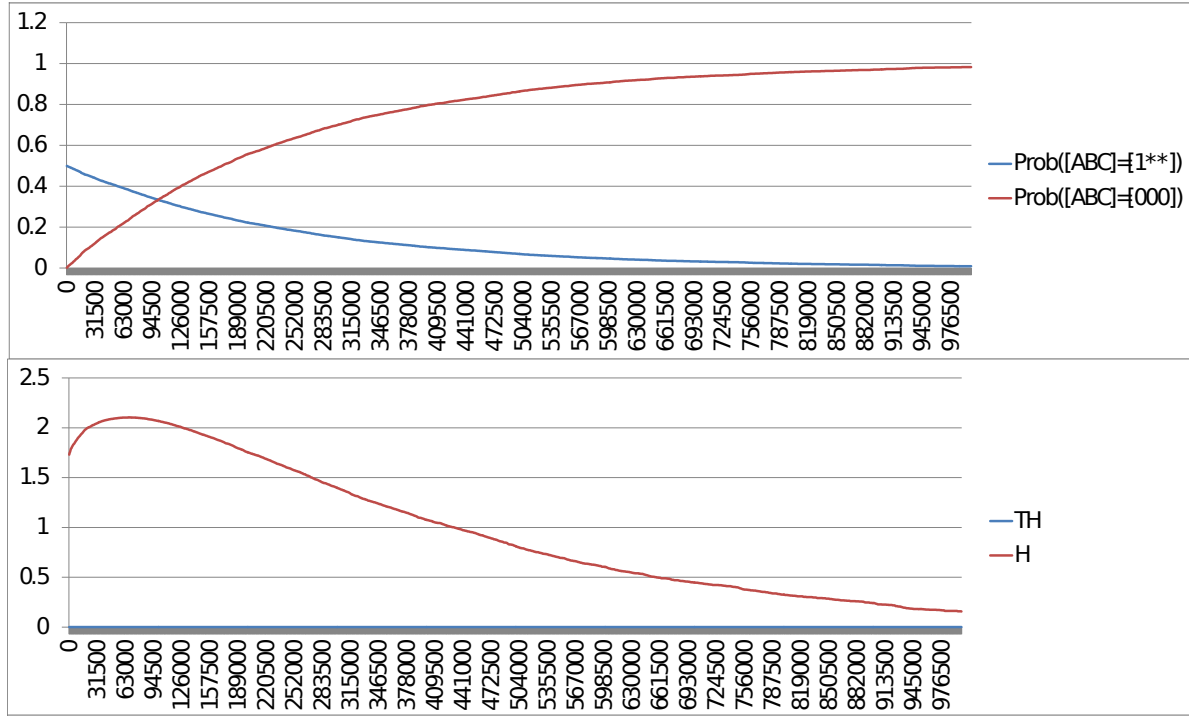
13

Figure 5: BKMC algorithm applied to the toy model, with a slow escape rate, plotted on a larger time scale. Time trajectory of probabilities ([ABC]=[000] and [ABC]=[1**]), the entropy (H) and the transition entropy (TH) are plotted. On a large time scale, the asymptotic behavior of both the entropy and the transition entropy is similar to the case of the fast escape rate (figure 3).
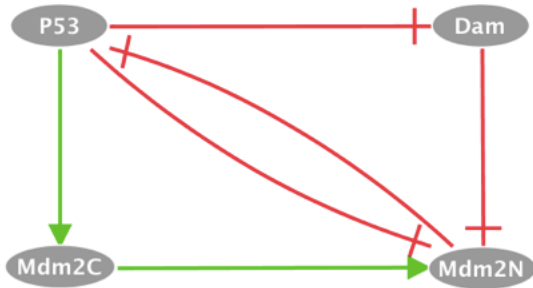

Figure 6: Boolean model of p53 response to DNA damage

allows to introduce logical rules and rates of node activation/inhibition in a flexible manner. This software provides global and semi-global outputs of the model dynamics that can be interpreted as signatures of the dynamical behaviors. These interpretations become particularly useful when the network state space is too large to be handled. The convergence of BKMC algorithm can be controlled by tuning different parameters: maximum time of the simulation, number of trajectories, length of a time window on which the average of probabilities is performed, and the threshold for the definition of stationary distributions clusters.

The next step is to apply BKMC algorithm with MaBoSS on large signaling networks, e.g. EGFR pathway, cell cycle signaling, apoptosis pathway, etc. The translation of existing Boolean models in MaBoSS is straightforward but requires the addition of transition rates. In these future works, we expect to illustrate
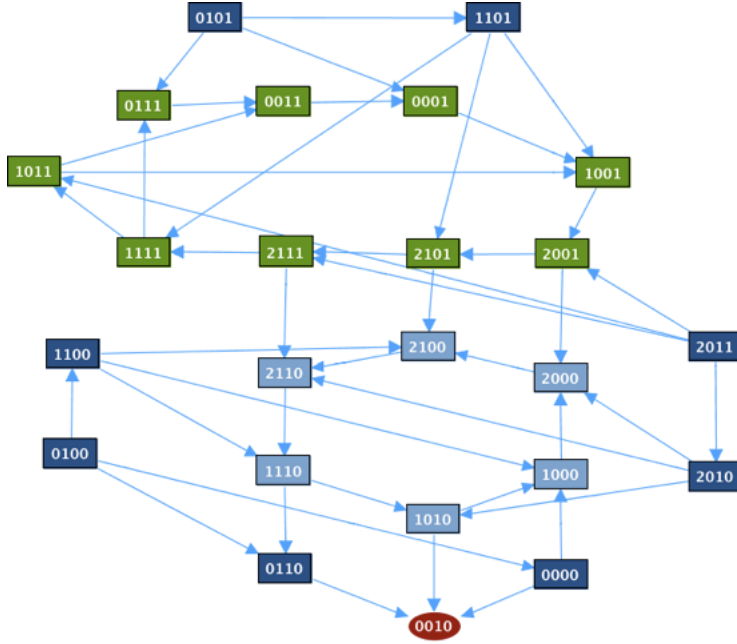
Figure 7: Transition graph of the p53 model.The node states should be read as [p53 Mdm2C Mdm2N Dam]
= [****] (where * can be either 0 or 1). For instance, [p53 Mdm2C Mdm2N Dam]=[1000] corresponds to a
state in which only p53 (at its level 1) is active. The nodes in green and the nodes in light blue belong to
two cycles, the node in red is the fixed point and the other state nodes are in dark blue.
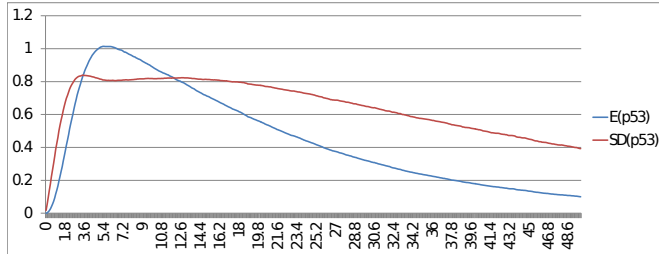

Figure 8: Time trajectories of p53 expectation value and standard deviation

the advantage of BKMC on other simulation algorithms. Moreover, in future developments of MaBoSS, we
plan to introduce methods for sensitivity analyses and to refine approximation methods used in BKMC, and
generalize Markov property.

Methods

BKMC generates stochastic trajectories. Here, we describe how we use and interpret them.

Network state probabilities on time window

To relate continuous time probabilities to real processes, an observable time window Δt is defined. A discrete
time ($\tau \in \mathbb{N}$) stochastic process $s(\tau)$ (that is **not** necessary Markovian) can be extracted from the continuous
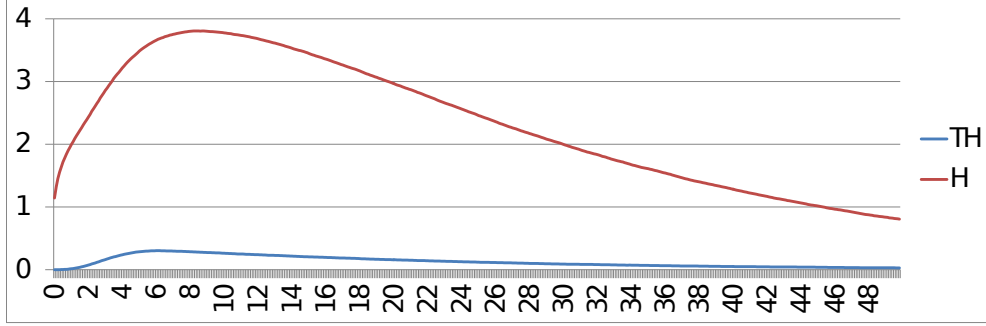
15

Figure 9: Time trajectories of the entropy (H) and the transition entropy (TH)

time Markov process:

$$\mathbf{P}\left[s(\tau) = \mathbf{S}\right] \equiv \frac{1}{\Delta t} \int_{\tau \Delta t}^{(\tau+1)\Delta t} dt\, \mathbf{P}\left[s(t) = \mathbf{S}\right] \tag{8}$$

BKMC is used for estimating $\mathbf{P}\left[s(\tau) = \mathbf{S}\right]$ as follows:

1. For each trajectory j, compute the time for which the system is in state $\mathbf{S}$, in the window $[\tau \Delta t, (\tau + 1)\Delta t]$. Divide this time by Δt. Obtain an estimate of $\mathbf{P}\left[s(\tau) = \mathbf{S}\right]$ for trajectory j, *i.e.* $\hat{\mathbf{P}}_j\left[s(\tau) = \mathbf{S}\right]$.

2. Compute the average over j of all $\hat{\mathbf{P}}_j\left[s(\tau) = \mathbf{S}\right]$ to obtain $\hat{\mathbf{P}}\left[s(\tau) = \mathbf{S}\right]$. Compute the error of this average ($\sqrt{\mathrm{Var}(\hat{\mathbf{P}}\left[s(\tau) = \mathbf{S}\right])}/\#$ trajectories).

Entropies

Once $\mathbf{P}\left[s(\tau) = \mathbf{S}\right]$ is computed, the entropy $H(\tau)$ can be estimated

$$H(\tau) = -\sum_{\mathbf{S}} \log_2\left(\mathbf{P}\left[s(\tau) = \mathbf{S}\right]\right) \mathbf{P}\left[s(\tau) = \mathbf{S}\right] \tag{9}$$

The entropy measures the disorder of the system. A maximum entropy means that all states have the same probability, a zero entropy means that one of the states has a probability of one. The estimation of the entropy can be seen as a global characterization (by a single real number) of a full probability distribution. The choice of $\log_2$ allows to interpret $H(\tau)$ in an easier manner: $2^{H(\tau)}$ is an estimate of the number of states that have a non-negligible probability in the time window $[\tau \Delta t, (\tau + 1)\Delta t]$. A more computer-like interpretation of $H(\tau)$ is the number of bits that is necessary for describing states of non-negligible probability.

The *Transition Entropy TH* is a finer measure that characterizes the system at the single trajectory level; it can be computed in the following way: for each state $\mathbf{S}$, there exists a set of possible transitions $\mathbf{S} \to \mathbf{S}'$. For each of these transitions, a probability is associated:

$$\mathbf{P}_{\mathbf{S} \to \mathbf{S}'} \equiv \frac{\rho_{\mathbf{S} \to \mathbf{S}'}}{\sum_{\mathbf{S}'} \rho_{\mathbf{S} \to \mathbf{S}'}}. \tag{10}$$

By convention, $\mathbf{P}_{\mathbf{S} \to \mathbf{S}'} = 0$ if there is no transition from $\mathbf{S}$ to any other state.

Therefore, the transition entropy TH can be associated to each state $\mathbf{S}$:

$$TH(\mathbf{S}) = -\sum_{\mathbf{S}'} \log_2(\mathbf{P}_{\mathbf{S} \to \mathbf{S}'}) \mathbf{P}_{\mathbf{S} \to \mathbf{S}'} \tag{11}$$

Similarly, $TH(\mathbf{S}) = 0$ if there is no transition from $\mathbf{S}$ to any other state. The *discrete time transition entropy* $TH(\tau)$ is defined as:

$$TH(\tau) = \sum_{\mathbf{S}} \mathbf{P}\left[s(\tau) = \mathbf{S}\right] TH(\mathbf{S})$$

This transition entropy is estimated in the following way:

1. For each trajectory j, compute the set (Φ) of visited states ($\mathbf{S}$) in time window $[\tau \Delta t, (\tau + 1)\Delta t]$ and their respective duration ($\mu_{\mathbf{S}}$). The estimated transition entropy is

$$\hat{TH}(\tau)_j = \sum_{\mathbf{S} \in \Phi} TH(\mathbf{S}) \frac{\mu_{\mathbf{S}}}{\Delta t} \tag{12}$$

2. Compute the average over j of all $\hat{TH}(\tau)_j$ to obtain $\hat{TH}(\tau)$. Compute the error of that average $(\sqrt{\mathrm{Var}(\hat{TH}(\tau))/\# \text{ trajectories}})$.

This transition entropy is a way to measure how deterministic the dynamics is. If the transition entropy is always zero, the system can only make a transition to a given state (although time of transitions remains stochastic).

If probability distributions on time window tend to constant values (or tend to a stationary distribution), the entropy and the transition entropy can help characterize this stationary distribution such that:

- A fixed point has zero entropy and zero transition entropy,

- A cyclic stationary distribution has non-zero entropy and zero transition entropy.

Entropy and transition entropy can be considered as "global characterization" of time evolution model: for a given time window, they always consist of two real numbers, whatever the size of the network.

Hamming distance distribution

The *Hamming Distance* between two states $\mathbf{S}$ and $\mathbf{S}'$ is the number of nodes that have different node states between $\mathbf{S}$ and $\mathbf{S}'$:

$$HD(\mathbf{S}, \mathbf{S}') \equiv \sum_i (1 - \delta_{S_i, S_i'}) \tag{13}$$

where δ is the Kronecker delta ($\delta_{S_i, S_i'} = 1$ if $S_i = S_i'$, $\delta_{S_i, S_i'} = 0$ if $S_i \neq S_i'$). Given a reference state $\mathbf{S}_{\mathrm{ref}}$, the Hamming distance distribution (over time) is given by

$$\mathbf{P}(HD, t) = \sum_{\mathbf{S}} \mathbf{P}\left[s(t) = \mathbf{S}\right] \delta_{HD, HD(\mathbf{S}, \mathbf{S}_{\mathrm{ref}})} \tag{14}$$

The estimation of discrete time Hamming distance distribution $\mathbf{P}(HD, \tau)$ is similar to that of stochastic probabilities on time window.

The Hamming distance distribution is a useful characterization when the set of instantaneous probabilities is compared to a reference state ($\mathbf{S}_{\mathrm{ref}}$). In that case, Hamming distance distribution describes how far this set is to this reference state.

The Hamming distance distribution can be considered as a "semi-global" characterization of time evolution: for a given time window, the size of this characterization is the number of nodes (to be compared with probabilities on time window, whose size is $2^{\#\mathrm{nodes}}$).

Initial conditions and outputs

Inputs Nodes are defined as the nodes for which the initial condition is fixed. Therefore, each trajectory of BKMC starts with fixed values of input nodes and random values of other nodes.

Internal nodes are nodes that are not considered for computing probability distributions, entropies and transition entropies. *Output nodes* are nodes that are not internal. Technically, probabilities are sum up over network states that differ only by the state of internal nodes. These internal nodes are only used for generating time trajectories through BKMC algorithm. Usually, nodes are chosen to be internal when the

corresponding species is not measured experimentally. Mathematically, it is equivalent to transform the original Markov process to a new stochastic process (that is not necessary Markovian) defined to a new network state space; this new state space is defined by states of output nodes. This raises the question of the transition entropy TH: formally, this notion has only a sense within Markovian processes, i.e. when there are no internal nodes. Here, we generalize the notion of transition entropy even in the case of internal nodes. Suppose that the system is in state $\mathbf{S}$:

- If the only possible transitions from state $\mathbf{S}$ to any other state consist of flipping an internal node, the transition entropy is zero.

- If there is, at least, one transition from state $\mathbf{S}$ to another state that flips a non-internal node, then only the non-internal nodes will be considered for computing probabilities in equations 10. In particular, $\sum_{\mathbf{S}'} \rho_{\mathbf{S} \to \mathbf{S}'}$ is computed only on non-internal node flipping events.

Reference nodes are nodes for which a reference node state is specified and for which the Hamming distance is computed. In this framework, a reference state is composed of reference nodes for which the node state is known and non-reference nodes for which the node state is unknown. Note that non-reference nodes may differ from internal nodes.

Stationary distribution characterization

It can be shown (see Supplementary Material, corollary 2, section 1.2) that instantaneous probabilities of a continuous time Markov process converge to a stationary distribution. Fixed points and cycles are two special cases of stationary distributions. They can be identified by the asymptotic behavior of entropy and transition entropy (this works only if no nodes are internal):

- if the transition entropy and the entropy converge to zero, then the process converges to a fixed point

- if the transition entropy and the entropy does not converge to zero, then the process converges to a cycle

More generally, the complete description of the Markov process asymptotic behavior can be expressed as a linear combination of the indecomposable stationary distributions.

A set of finite trajectories, produced by BKMC, can be used to estimate the set of indecomposable stationary distributions. Consider a trajectory $\hat{\mathbf{S}}(t), t \in [0, T], i = 1, \cdots, n$. Let $I_{\mathbf{S}}(t) \equiv \delta_{\mathbf{S}, \hat{\mathbf{S}}(t)}$. The estimation of the associated indecomposable stationary probability distribution (s_0) is done by averaging over the whole trajectory:

$$\hat{\mathbf{P}}\left[s_0 = \mathbf{S}\right] = \frac{1}{T} \int_0^T dt I_{\mathbf{S}}(t) \tag{15}$$

Therefore, a set of indecomposable stationary distribution estimates can be obtained by a set of trajectories. These indecomposable stationary distribution estimates should be clustered in groups, where each group consists of estimates for the same indecomposable stationary distribution. For that, we use the fact that two indecomposable stationary distributions are identical if they have the same support, *i.e.* the same set of network states with non-zero probabilities (shown in Supplementary Material, theorem 2). Therefore, it is possible to quantify how similar two indecomposable stationary distribution estimates are. A *similarity coefficient* $D(s_0^{(i)}, s_0^{(j)}) \in [0, 1]$, given two stationary distribution estimates $s_0^{(i)}$ and $s_0^{(j)}$, is defined:

$$D(s_0^{(i)}, s_0^{(j)}) \equiv \left(\sum_{\mathbf{S} \in \text{support}(s_0^{(i)}, s_0^{(j)})} \hat{\mathbf{P}}\left[s_0^{(i)} = \mathbf{S}\right] \right) \left(\sum_{\mathbf{S}' \in \text{support}(s_0^{(i)}, s_0^{(j)})} \hat{\mathbf{P}}\left[s_0^{(j)} = \mathbf{S}'\right] \right) \tag{16}$$

where

$$\text{support}(s_0^{(i)}, s_0^{(j)}) \equiv \left\{ \mathbf{S} \text{ such that } \hat{\mathbf{P}}\left[s_0^{(i)} = \mathbf{S}\right] \hat{\mathbf{P}}\left[s_0^{(j)} = \mathbf{S}\right] > 0 \right\} \tag{17}$$

Clusters can be constructed when a similarity threshold α is provided. A cluster of stationary distributions is defined as follows:

$$\mathcal{C} = \{s_0 | \exists s_0' \in \mathcal{C} \text{ s. t. } D(s_0, s_0') \geq \alpha\} \tag{18}$$

For each cluster $\mathcal{C}$, an associated distribution $s_{\mathcal{C}}$, that estimates an indecomposable stationary distribution, can be defined

$$\mathbf{P}\left[s_{\mathcal{C}} = \mathbf{S}\right] = \frac{1}{|\mathcal{C}|} \sum_{s \in \mathcal{C}} \mathbf{P}\left[s = \mathbf{S}\right] \tag{19}$$

Errors on this estimate can be computed by

$$\mathrm{Err}\left(\mathbf{P}\left[s_{\mathcal{C}} = \mathbf{S}\right]\right) = \sqrt{\mathrm{Var}(\mathbf{P}\left[s = \mathbf{S}\right], s \in \mathcal{C})/|\mathcal{C}|} \tag{20}$$

Notice that this clustering procedure has no sense if the process is not Markovian; therefore, no node will be considered as internal.

List of abbreviations

BKMC: Boolean Kinetic Monte-Carlo
AT: Asynchronous transition
ODEs: Ordinary Differential Equations
MaBoSS: Markov Boolean Stochastic Simulator

Author's contributions

G. Stoll organized the project, set up the algorithms, participated in writing the software, set up the examples and wrote the article.
    E. Viara wrote the software and participated in setting up the algorithms.
    E. Barillot participated in writing the article.
    L. Calzone organized the project, set up the examples and wrote the article.

Acknowledgements

References

[1] Kauffman S: **Homeostasis and differentiation in random genetic control networks**. *Nature* 1969, **224**:177–178.

[2] Thomas R: **Regulatory networks seen as asynchronous automata: a logical description**. *Journal of Theoretical Biology* 1991, **153**:1–23.

[3] Stoll G, Rougemont J, Naef F: **Few crucial links assure checkpoint efficiency in the yeast cell-cycle network**. *Bioinformatics* 2006, **22**(20):2539–2546.

[4] Stoll G, Bischofberger M, Rougemont J, Naef F: **Stabilizing patterning in the Drosophila segment polarity network by selecting models in silico**. *Biosystems* 2010, **102**:3–10.

[5] Garg A, Mohanram K, Di Cara A, De Micheli G, Xenarios I: **Modeling stochasticity and robustness in gene regulatory networks**. *Bioinformatics* 2009, **25**(12):i101–i109.

[6] Fauré A, Naldi A, Chaouiya C, Thieffry D: **Dynamical analysis of a generic Boolean model for the control of the mammalian cell cycle**. *Bioinformatics* 2006, **22**(14):e124–e131.

[7] Shmulevich I, Dougherty E, Kim S, Zhang W: **Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks**. *Bioinformatics* 2002, **18**(2):261.

[8] Gonzalez A, Naldi A, Sanchez L, Thieffry D, Chaouiya C: **GINsim: A software suite for the qualitative modelling, simulation and analysis of regulatory networks**. *Biosystems* 2006, **84**(2):91–100.

[9] Wunderlich Z, DePace A: **Modeling transcriptional networks in Drosophila development at multiple scales**. *Current Opinion in Genetics & Development* 2011.

[10] MacArthur B, Ma'ayan A, Lemischka I: **Systems biology of stem cell fate and cellular reprogramming**. *Nature Reviews Molecular Cell Biology* 2009, **10**(10):672–681.

[11] Saez-Rodriguez J, Alexopoulos L, Zhang M, Morris M, Lauffenburger D, Sorger P: **Comparing signaling networks between normal and transformed hepatocytes using discrete logical models**. *Cancer research* 2011, **71**(16):5400.

[12] Siebert H, Bockmayr A: **Temporal constraints in the logical analysis of regulatory networks**. *Theoretical Computer Science* 2008, **391**(3):258–275.

[13] Öktem H, Pearson R, Yli-Harja O, Nicorici D, Egiazarian K, Astola J, et al.: **A computational model for simulating continuous time Boolean networks**. In *Proceedings of IEEE International Workshop on Genomic Signal Processing and Statistics (GENSIPS'02)* 2002.

[14] Vahedi G, Faryabi B, Chamberland J, Datta A, Dougherty E: **Sampling-rate-dependent probabilistic Boolean networks**. *Journal of theoretical biology* 2009, **261**(4):540–547.

[15] Sevim V, Gong X, Socolar J: **Reliability of transcriptional cycles and the yeast cell-cycle oscillator**. *PLoS computational biology* 2010, **6**(7):e1000842.

[16] Teraguchi S, Kumagai Y, Vandenbon A, Akira S, Standley D: **Stochastic binary modeling of cells in continuous time as an alternative to biochemical reaction equations**. *Physical Review E* 2011, **84**(6):062903.

[17] Bauer A, Jackson T, Jiang Y, Rohlf T: **Receptor cross-talk in angiogenesis: Mapping environmental cues to cell phenotype using a stochastic, Boolean signaling network model**. *Journal of Theoretical Biology* 2010, **264**(3):838–846.

[18] Abou-Jaoudé W, Ouattara D, Kaufman M: **From structure to dynamics: Frequency tuning in the p53-Mdm2 network:: I. Logical approach**. *Journal of theoretical biology* 2009, **258**(4):561–577.

[19] Van Kampen N: **Stochastic Processes in Physics and Chemistry. 2004**.

[20] Shiryaev A: **Probability, volume 95 of Graduate texts in mathematics** 1996.

[21] Chaves M, Albert R, Sontag E: **Robustness and fragility of Boolean models for genetic regulatory networks**. *Journal of theoretical biology* 2005, **235**(3):431–449.

[22] Chaouiya C: **Petri net modelling of biological networks**. *Briefings in Bioinformatics* 2007, **8**(4):210–219.

[23] Young W, Elcock E: **Monte Carlo studies of vacancy migration in binary ordered alloys: I**. *Proceedings of the Physical Society* 1966, **89**:735.

[24] Gillespie D: **A general method for numerically simulating the stochastic time evolution of coupled chemical reactions**. *Journal of computational physics* 1976, **22**(4):403–434.